Concept Pointer Network for Abstractive Summarization

Wang Wenbo[1], **Gao Yang**[1,2*], **Huang Heyan**[1,2] **and Zhou Yuxiang**[1]
[1] School of Computer Science and Technology, Beijing Institute of Technology
[2] Beijing Engineering Research Center of High Volume Language Information Processing and Cloud Computing Applications
gyang@bit.edu.cn

Abstract

A quality abstractive summary should not only copy salient source texts as summaries but should also tend to generate new conceptual words to express concrete details. Inspired by the popular pointer generator sequence-to-sequence model, this paper presents a concept pointer network for improving these aspects of abstractive summarization. The network leverages knowledge-based, context-aware conceptualizations to derive an extended set of candidate concepts. The model then points to the most appropriate choice using both the concept set and original source text. This joint approach generates abstractive summaries with higher-level semantic concepts. The training model is also optimized in a way that adapts to different data, which is based on a novel method of distantly-supervised learning guided by reference summaries and testing set. Overall, the proposed approach provides statistically significant improvements over several state-of-the-art models on both the DUC-2004 and Gigaword datasets. A human evaluation of the model's abstractive abilities also supports the quality of the summaries produced within this framework.

1 Introduction

Abstractive summarization (ABS) has gained overwhelming success owing to a tremendous development of sequence-to-sequence (seq2seq) model and its variants (Rush et al., 2015; Chopra et al., 2016; Paulus et al., 2017; Guo et al., 2018; Gao et al., 2019). In tandem with seq2seq models, pointer generator was developed by See et al. (2017) as a solution to tackle the rare words and out-of-vocabulary (OOV) problem associated with generative-based models. The idea behind is to use attention as a pointer to determine the probability of generating a word from both a vocabulary distribution and the source text. Pointer generator networks have also been extensively accepted by the ABS community due to their efficacy with long document summaries (Chen and Bansal, 2018; Hsu et al., 2018), title summarization (Sun et al., 2018), etc.

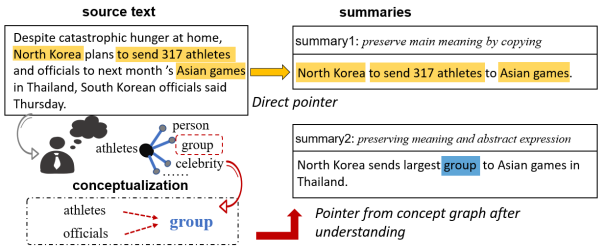


Figure 1: "summary1" only copies keyword from the source text, while "summary2" generates new concepts to convey the meaning.

However, the current power of abstractive summarization falls short of their potential. As the example in Figure 1 shows, a seq2seq model with a pointer mechanism (marked as the direct pointer) is likely to merely copy parts of the original text to form a summary using keywords and phrases, such as "317 athletes". Conversely, a more human-like summary would be based on one's own understanding of the detail in the words, expressed as higher-level concepts drawn from world knowledge—like using the word "group" to replace "athletes and officials". This indicates that a good summary should not simply copy original material, it should also generate new and even abstract concepts that reflect high-level semantics.

Therefore, a pointer generator network that solely considers the source material to generate a summary does not adequately satisfy the needs of high-quality abstractive summarization. We argue that concepts have a greater ability to express deeper meanings than verbatim words. As such, it is essential to explore the potential of us-

* Corresponding author

Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, pages 3067–3076, Hong Kong, China, November 3–7, 2019.

ing concepts from world knowledge to assist with abstractive summarization. Our developed model not only points to informative source texts but also leverages conceptual words from human knowledge in the summaries it generates.

Hence, in this paper, we propose a novel model based on a concept pointer generator that encourages the generation of conceptual and abstract words. As a hidden benefit, the model also alleviates the OOV problems. Our model uses pointer network to capture the salient information from a source text, and then employs another pointer to generalize the detailed words according to their upper level of expressions. Finally, the output is also consistent with language model by the seq2seq generator. Unique to our concept pointer is a set of concept candidates particular for a word that is drawn from a huge knowledge base. The set of candidates adheres to a concept distribution, where the probability of each concept being generated is linked to how strongly the candidate represents each word. Moreover, the concept distribution is iteratively updated to better explain the target word given the context of the source material and inherent semantics in the texts. Hence, the learned concept pointer points to the most suitable and expressive concepts or words. The optimization function is adaptive so as to cater for different datasets with distantly-supervised training. The network is then optimized end-to-end using reinforcement learning, with the distant-supervision strategy as a complement to further improve the summary.

Overall, the contributions of this paper are: 1) a novel concept pointer generator network that leverages context-aware conceptualization and a concept pointer, both of which are jointly integrated into the generator to deliver informative and abstract-oriented summaries; 2) a novel distant supervision training strategy that favors model adaptation and generalization, which results in performance that outperforms the well-accepted evaluation-based reinforcement learning optimization on a test-only dataset in terms of ROUGE metrics; 3) a statistical analysis of quantitative results and human evaluations from comparative experiments with several state-of-the-art models that shows the proposed method provides promising performance.

2 Related Work

Abstractive summarization supposedly digests and understands the source content and, consequently, the generated summaries are typically a reorganization of the wording that sometimes form new sentences. Historically, abstractive summarization has been performed through rule-based sentence selection (Dorr et al., 2003), key information extraction (Genest and Lapalme, 2011), syntactic parsing (Bing et al., 2015) and so on. However, more recently, seq2seq models with attention have played a more dominant role in generating abstractive summaries (Rush et al., 2015; Chopra et al., 2016; Nallapati et al., 2016; Zhou et al., 2017). Extensions to the seq2seq approach include an intra-decoder attention (Paulus et al., 2017) and coverage vectors (See et al., 2017) to decrease repetition in phrasing. Copy mechanism (Gu et al., 2016) has been integrated into these models to tackle OOV problem. Zhou et al. (2018) went on to propose SeqCopyNet which copies complete sequences from an input sentence to further maintain the readability of the generated summary.

Pointer mechanism (Vinyals et al., 2015) has drawn much attention in text summarization (See et al., 2017), because this technique not only provides a potential solution for rare words and OOV but also extends abstractive summarization in a flexible way (Çelikyilmaz et al., 2018). Further, pointer generator models can effectively adaptive to both extractor and abstractor networks (Chen and Bansal, 2018), and summaries can be generated by incorporating a pointer-generator and multiple relevant tasks (Guo et al., 2018), such as question or entailment generation, or multiple source texts (Sun et al., 2018).

However, work particularly targets the problem of the abstraction is rare. Abstract Meaning Representation (AMR) is used to transform a sentence into a concept graph, then merge those similar concept nodes to form a new summary graph (Liu et al., 2018). Concepts are also incorporated as auxiliary features (Guo et al., 2017). Kryscinski et al. (2018) and Weber et al. (2018) define the number of new n-grams as the primary criteria of abstractiveness. This makes sense in most cases. But, we believe that abstraction means summarizing detailed content with higher-level semantically related concepts, which has motivated the development of the model proposed in this paper.

3 The Proposed Model

Neural abstractive summarization can be described as a generation process where a sequential input is summarized into a shorter sequential output through a neural network. Suppose that the sequential input $\boldsymbol{x} = \{x_1, \ldots, x_i, \ldots, x_n\}$ is a sequence of n number of words, and i is the index of the input. The shorter (i.e., summarized) sequence of output is denoted as $\boldsymbol{y} = \{y_1, \ldots, y_t, \ldots, y_m\}$ with number of m words, and t indicates a time step. As Figure 2 shows, our model consists of two sub-modules —an encoder-decoder module and the proposed concept pointer generator module.

3.1 Encoder-Decoder Framework

This process is formulated as an encoder-decoder framework that consists of an encoder and an attention-equipped decoder. We use a two-layer bi-directional LSTM-RNN encoder and one-layer uni-directional LSTM-RNN decoder along with attention mechanism.

Formally, the encoder produces sequential hidden states as $(\overrightarrow{\boldsymbol{h}}_1, \ldots, \overrightarrow{\boldsymbol{h}}_n)$ and $(\overleftarrow{\boldsymbol{h}}_1, \ldots, \overleftarrow{\boldsymbol{h}}_n)$ in the corresponding positions, and the bi-directional $h_i = f_{LSTM}(h_{i-1}, x_i)$. Each word x_i in the sequence can be represented as a concatenation of the bi-directional hidden states, i.e., $\boldsymbol{h}_i = [\overrightarrow{\boldsymbol{h}}_i, \overleftarrow{\boldsymbol{h}}_i]$. The decoder generates a target summary from a vocabulary distribution $P_{vocab}(w)$, which is based on a context vector $\boldsymbol{h}_t^*$ through the following process:

$$\begin{aligned} P_{vocab}(w) &= P(y_t|\boldsymbol{y}_{<t}, \boldsymbol{x}; \theta) \\ &= \text{sfm}(\boldsymbol{W}_2(\boldsymbol{W}_1[\boldsymbol{s}_t, \boldsymbol{h}_t^*] + \boldsymbol{b}_1) + \boldsymbol{b}_2) \end{aligned} \tag{1}$$

where $\boldsymbol{s}_t$ is the hidden state of the decoder at time step t , and $\boldsymbol{h}_t^*$ is the context vector at time step t. $\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{b}_1, \boldsymbol{b}_2$ are trainable parameters, and sfm$(\cdot)$ is short for softmax function.

The context vector $\boldsymbol{h}_t^*$ is computed by a weighted sum of the hidden representations of the source text, and the weight is denoted as the attention $\alpha_{t,i}$.

$$\begin{aligned} \boldsymbol{h}_t^* &= \sum\nolimits_{i=1}^{n} \alpha_{t,i} \boldsymbol{h}_i \\ \alpha_{t,i} &= \text{sfm}(\boldsymbol{v}^\top \tanh(\boldsymbol{W}_h \boldsymbol{h}_i + \boldsymbol{W}_s \boldsymbol{s}_t + \boldsymbol{b})) \end{aligned} \tag{2}$$

The softmax function normalizes the vector of a distribution over the input position, and $\boldsymbol{v}, \boldsymbol{W}_h, \boldsymbol{W}_s, \boldsymbol{b}$ are trainable parameters.

3.2 Concept Pointer Generator

Pointer networks use attention as a pointer to select segments of the input as outputs (Vinyals et al., 2015). As such, a pointer network is a suitable mechanism for extracting salient information, while remaining flexible enough to interface with a seq2seq model for generating an abstractive summarization (See et al., 2017). Our proposed model is essentially an upgrade to this configuration that integrates a new concept pointer network within a unified framework.

3.2.1 Context-aware Conceptualization

"Understanding" the instances of a word requires a taxonomic knowledge base that relates those words to a concept space. In our model, we use an isA taxonomy, called the Microsoft Concept Graph[1] (Wang et al., 2015), to serve this purpose for two reasons (Wang and Wang, 2016). First, this graph provides a huge concept space with multi-word terms that cover concepts of worldly facts as concepts, instances, relationships, and values[2]. Second, the relationships between concepts and entities are probabilistic as a measure of how strongly they are related. Moreover, the probabilities are trustworthy given they have been derived from evidence found in billions of webpages, search log data, and other existing taxonomies. Our model is data-driven and, therefore, is more easily adaptable with probabilities. All these characteristics make the Microsoft Concept Graph a suitable choice for our model. More detailed examples are available in Appendix A.

The concept graph specifies the probability that each instance x belongs to a concept c, $p(c|x)$. Given a word x, we have a distribution over a set of related concepts. Yet, this raises the question of how to identify a context-appropriate concept for a word from the distributional set of candidate concepts. For instance, *apple* in the context of "*an apple is good for you health*" tends to be associated with the concept of *fruit* instead of *company*. Formally, given a word x_i in a training sentence, a set of k concept candidates, $C_i = \{c_i^1, c_i^2, \cdots, c_i^k\}$, is linked to the word from the knowledge base, with distributional probabilities over the concepts, i.e.,

[1]The Microsoft concept graph was derived from Probase project. The public data can be downloaded via the provided API: https://concept.research.microsoft.com/Home/API

[2]The current version is mined from billions of web pages, containing 5.3 M unique concepts, 12.5 M unique entities and 85 M isA relations.

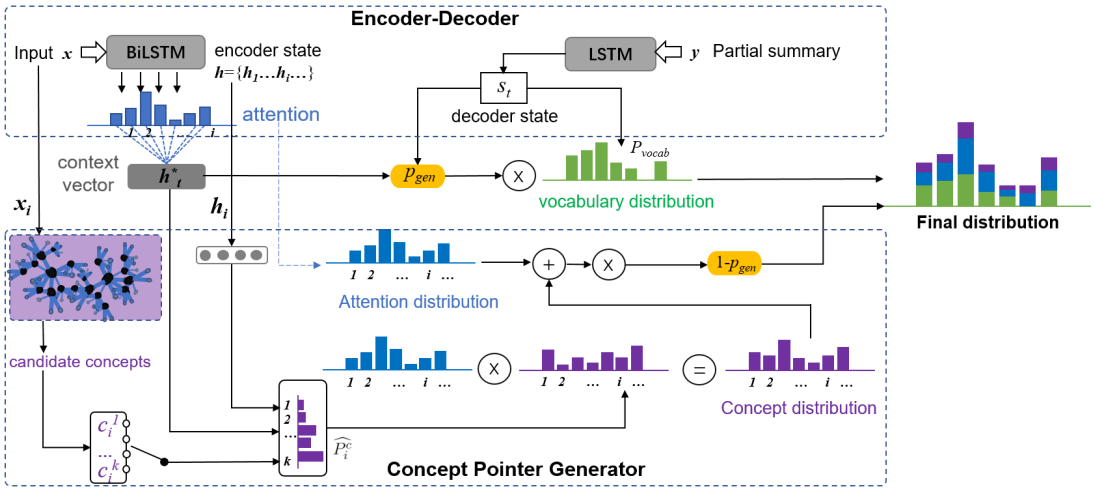


Figure 2: The architecture of our model. Blue bar represents the attention distribution over the inputs. Purple bar represents the concept distribution over the inputs. Noted that, this distribution can be sparse since not every word has its upper concept. Green bar represents the vocabulary distribution generated from seq2seq component.

$P(C|x_i) = \{p(c_i^1), p(c_i^2), \cdots, p(c_i^k)\}$. The task is to find the most suitable concept c_i^j to fit the updated context, represented by the vector $\boldsymbol{h}_t^*$ in Equation (2), at each time step t.

In the case of generating summaries given updated contexts, a weighted update of the distributional concept candidates needs to be performed. In the model, the updated weight, denoted as β_i^j, is estimated by a softmax classifier that is jointly conditioned on the hidden representation of the word $\boldsymbol{h}_i$, the context vectors $\boldsymbol{h}_t^*$, and each of concept vectors:

$$\beta_i^j = \text{softmax}(\boldsymbol{W}_{h'}[\boldsymbol{h}_i, \boldsymbol{h}_t^*, \boldsymbol{c}_i^j]) \quad (3)$$

where $j \in [1, k]$, $\boldsymbol{W}_{h'}$ is a trainable parameter, and $\boldsymbol{c}_i^j$ is the vector of the jth concept candidate, which is a representation of the input embeddings.

Together with the concept probability from the existing knowledge base $p(c_i^j)$ and the updated weights based on the contexts β_i^j, a context-aware conceptualized probability of jth concept for the word x_i, $P_{i,j}^c$, is finally estimated as

$$P_{i,j}^c = p(c_i^j) + \gamma \beta_i^j \quad (4)$$

where γ is a tunable parameter. Theoretically, we will end up with a number of k relevant concepts for each word $C_i = \{c_i^1, \cdots, c_i^k\}$ with a probability distribution over the set, which is learned as $P_i^c = \{P_{i,1}^c, \cdots, P_{i,j}^c, \cdots, P_{i,k}^c\}$.

3.2.2 Concept Pointer Generator

The basic pointer generator network contains two sub-modules, one is the pointer network and the other is the generation network. These two sub-modules jointly determine the probabilities of the words in the final generated summary. The generation probability p_{gen} for the generation network (See et al., 2017) is learned by

$$p_{gen} = \sigma(\boldsymbol{W}_{h^*}\boldsymbol{h}_t^* + \boldsymbol{W}_s\boldsymbol{s}_t + \boldsymbol{W}_y\boldsymbol{y}_{t-1} + \boldsymbol{b}_{gen}) \quad (5)$$

where σ is a sigmoid function.

For the pointer network, our model consists of a pointer to the source text and a further concept pointer to the relevant concepts that have arisen from the source content. These two separate pointers are calculated as follows. The first pointer is taken based on the attention distribution $\alpha_{t,i}$ over the source text. The second concept pointer is operated over a concept distribution of the source text that is scaled element-wise by the attention distribution.

To train the model, given the likelihood of each concept in the current context, the updates could be performed in two ways. In a hard assignment, the concept that receives the highest score would be selected for the update:

$$\widehat{P_i^c}_{\text{argmax}} = P_{i,a}^c, \text{where } a = \arg\max_j(\beta_i^j) \quad (6)$$

where a is the index of maximized generated weight based on the contexts, and $P_{i,a}^c$ is obtained by Eqs. (4).

In random selection, each of the concept candidates could be trained randomly to update the parameters:

$$\widehat{P_i^c}_{\text{random}} = P_{i,j}^c \sim P_i^C \quad (7)$$

where j represents the selected concept index. Considering the above baseline generation network and both the pointer networks, our final output distribution is

$$P_{\text{final}}(w) = p_{gen}P_{vocab}(w) + (1 - p_{gen}) \\ (\sum_{i:w_i=w} \alpha_{t,i} + \sum_{i:w_i=w} \alpha_{t,i} \times \widehat{P_i^c}) \quad (8)$$

where $\widehat{P_i^c}$ can be updated by $\widehat{P_{i\ \arg\max}^c}$, or $\widehat{P_{i\ \text{random}}^c}$. The difference between these two choices is demonstrated in the Experiments section.

3.3 Objective Learning

3.3.1 Basic MLE

The baseline objective is derived by maximizing the likelihood training for the seq2seq generation, given a reference summary $y^* = \{y_1^*, y_2^*, \cdots, y_{m'}^*\}$ for document x. The training objective is to minimize the negative log-likelihood of the target word sequence:

$$\mathcal{L}_{MLE} = -\sum_{t=1}^{m'} \log P(y_t^*|y_1^*, \cdots, y_{t-1}^*, \boldsymbol{x}) \quad (9)$$

3.3.2 Evaluation based Reinforcement Learning (RL)

Similar to Paulus et al. (2017), policy gradient methods can directly optimize discrete target evaluation metrics, such as ROUGE. The basic idea is to explore new sequences and compare them to the best greedy baseline sequence. Once the baseline sequence $\hat{y}$, or sampled sequence y^s, are generated, they are compared against the reference sequence y^* to compute the rewards $r(\hat{y})$ and $r(y^s)$, respectively. In the RL training stage, two separate output candidates at each time step are produced: y^s is sampled from the probability distribution $P(y_t^s|y_1^s, \cdots, y_{t-1}^s, x)$, and $\hat{y}$ is the baseline output. The training objective is then to minimize

$$\mathcal{L}_{RL} = (r(\hat{y}) - r(y^s)) \sum_{t=1}^{m'} \log P(y_t^s|y_1^s, \cdots, y_{t-1}^s, x) \quad (10)$$

It is noteworthy that the samples y^s are selected from a wide range of vocabularies extended by all the concept candidates. This strategy ensures that the model learns to generate sequences with higher rewards by better exploring a set of close concepts.

Thus, the combination of these two objectives yield improved task-specific scores while catering a better language model: $\mathcal{L}_{final} = \lambda\mathcal{L}_{RL} + (1 - \lambda)\mathcal{L}_{MLE}$, where λ is a soft-switch between the two objectives. The model is pre-trained with MLE loss, then switch to the final loss.

3.3.3 Distant Supervision (DS) for Model Adaption

Our intuition is that, if the summary-document pairs are dissimilar to the testing set, the model could be retrained to adapt to weaken the influence of the dissimilarity on the final loss. The result would be a training model that better fits the specific testing data. The challenge is that there are no explicit supervision labels to indicate whether the training set is close to the testing set, so a new training paradigm is needed. In answer to this need and also to provide end-to-end functionality in the model, we developed a simple approach for labeling summary-document pairs by calculating the Kullback-Leibler (KL) divergence between each training reference summary and a set of testing documents. In this way, the training pairs are distantly-labelled for training the model.

Specifically, the representations of the reference summaries and the testing set are computed by summing all the involved word embeddings. Given a testing document x_l^{d}, where $l \in [1, N^{\text{d}}]$ and N^{d} is the size of the testing corpus, the vector-based representation of one document is $\boldsymbol{x}_l^{\text{d}} = \exp(\sum_{i=1}^{n'} \boldsymbol{x}_i^{\text{d}})$, where n' is the number of document words involved. The reference summary is represented by $\boldsymbol{y^*} = \exp(\sum_{t=1}^{m'} \boldsymbol{y_t^*})$. We normalize these vectors through a softmax function to cater for KL calculation. The model adaption with the distant labels is defined as:

$$\mathcal{L}_{DS} = (\pi - \tfrac{1}{N^{\text{d}}} \sum_{l=1}^{N^{\text{d}}} D_{\text{KL}}(\boldsymbol{y^*}, \boldsymbol{x}_l^{\text{d}}))\mathcal{L}_{\text{MLE}} \quad (11)$$

where $D_{\text{KL}}(.)$ indicates the KL divergence between y^* and x_l^{d}, and π is a constant parameter that is tuned via adaption to the testing set. The divergences are averaged within the testing set, which indicates the overall distances between testing set and each of the reference summary-document pairs. In this way, the samples in the training corpus are distantly annotated as either relevant or irrelevant for model adaption, noting that the model is pre-trained with the MLE loss before switching to distantly-supervised training.

4 Experiments

Datasets: To evaluate the effectiveness of our proposed model, we conducted training and test-

Table 1: ROUGE F1 evaluation results on the Gigaword and ROUGE recall on DUC-2004 test set. The results with † mark are taken from the corresponding papers. Underlined scores are the best without additional optimization. Bold scores are the best between the two optimization strategies. ⋆ mark indicates the improvements from the baselines to the concept pointer are statistically significant using a two-tailed t-test ($p < 0.01$).

Models	Gigaword			DUC-2004		
	RG-1	RG-2	RG-L	RG-1	RG-2	RG-L
ABS+† (Rush et al., 2015)	29.76	11.88	26.96	28.18	8.46	23.81
Luong-NMT† (Luong et al., 2015)	33.10	14.45	30.71	28.55	8.79	24.43
RAS-Elman† (Chopra et al., 2016)	33.78	15.97	31.15	28.97	8.26	24.06
lvt5k-lsent† (Nallapati et al., 2016)	35.30	16.64	32.62	28.61	9.42	25.24
SEASS† (Zhou et al., 2017)	36.15	17.54	33.63	29.21	9.56	25.51
Seq2seq+att ⋆(our impl.)	33.11	14.67	31.06	28.57	9.31	24.81
Pointer-generator ⋆(our impl.) (See et al., 2017)	35.98	15.99	33.33	28.28	10.04	25.69
Pointer-Cov.-Entail.-Quest.† (Guo et al., 2018)	35.98	17.76	33.63	-	-	-
Seq2seq-Sel.-MTL-ERAM† (Li et al., 2018)	35.33	17.27	33.19	29.33	10.24	25.24
CGU† (Lin et al., 2018)	36.3	18.0	33.8	-	-	-
Concept pointer	36.62	16.40	33.98	29.17	10.38	26.34
Concept pointer+RL	**38.02**	16.97	**35.43**	29.34	9.84	26.60
Concept pointer+DS	37.01	**17.10**	34.87	**30.39**	**10.78**	**27.53**

ing on two popular datasets. The first was the English Gigaword Fifth Edition corpus (Parker et al., 2011). We replicated the pre-processing steps in (Rush et al., 2015) to obtain the same training/testing data. After pre-processing, the corpus contained about 3.8M sentence-summary pairs as training set and 189K pairs as the development set. Once pairs with empty titles were removed, the testing set numbered 1951 pairs. The second dataset, DUC2004, was only used for testing. This dataset consists of 500 document-headline summary pairs, where each document is paired with four reference summaries written by humans.

Evaluation Metrics: We used ROUGE (Lin, 2004) as the evaluation metric, which measures the quality of a summary by computing the overlapping lexical elements between the candidate summary and a reference summary. Following previous practice, we assessed RG-1 (unigram), RG-2 (bigram) and RG-L (longest common subsequence - LCS). Noted that the English Gigaword[3] testing set contains references of different lengths, while the DUC-2004[4] testing set fixes the summary length to 75 bytes.

Training Setups: We initialize word embeddings with 128-d vectors and fine-tune them during training. Concepts share the same embeddings with the words. The vocabulary size was set to 150k for both the source and target text. The hidden state size was set to 256. The vocabulary size is increased from around 602 to 2216 concepts w.r.t the different number ($k = 1, \cdots, 5$) of concept candidates for each word. Note that the generated concepts with UNKs were subsequently deleted. Our code is available on `https://github.com/wprojectsn/codes`, and the vocabularies and candidate concepts are also included.

We trained our models on a single GTX TITAN GPU machine. We used the Adagrad optimizer with a batch size of 64 to minimize the loss. The initial learning rate and the accumulator value were set to 0.15 and 0.1, respectively. We used gradient clipping with a maximum gradient norm of 2. At the time of decoding, the summaries were produced through a beam search of size 8. The hyper-parameter settings were $\lambda = 0.99$, $\gamma = 0.1$, $\pi = 2.92$ on DUC-2004 and $\pi = 1.68$ on Gigaword. We trained our concept pointer generator for 450k iterations yielded the best performance, then took the optimization using RL rewards for RG-L at 95K iterations on DUC-2004 and at 50K iterations on Gigaword. We took the distance-supervised training at 5K iterations on DUC-2004 and at 6.5K iterations on Gigaword.

Baselines: The following state-of-the-art baselines were used as comparators. **ABS+** (Rush

[3]The ROUGE evaluation option is, -m -n 2 -s

[4]The ROUGE evaluation option is, -n 2 -m -b 75 -s

Table 2: OOV problem analysis: percentages (%(NO. UNK/NO. all generated words)) of generating UNK w.r.t the following three models on Gigaword and DUC-2004 datasets.

Method	Gigaword	DUC-2004
seq2seq+att	4.02%(570/14183)	2.08%(85/4079)
Pointer generator	1.16%(207/17859)	0.31% (16/5140)
Concept pointer	1.12%(202/17950)	0.23%(12/5230)

Table 3: Abstractiveness: percentage of novel n-grams on Gigaword dataset.

Models	Novel n-gram (%)		
	1-gram	2-gram	3-gram
Pointer generator	14.3	41.9	63.4
Concept pointer	17.2	45.8	68.5
Reference summary	25.4	65.6	78.4

et al., 2015) is a tuned ABS model with additional features. **Luong-NMT** (Luong et al., 2015) is a two-layer LSTM encoder-decoder. **RAS-Elman** (Chopra et al., 2016) is a convolution encoder and an Elman RNN decoder with attention. **Seq2seq+att** is two-layer BiLSTM encoder and one-layer LSTM decoder equipped with attention. **lvt5k-lsent** (Nallapati et al., 2016) uses temporal attention to keep track of the past attentive weights of the decoder and restrains the repetition in later sequences. **SEASS** (Zhou et al., 2017) includes an additional selective gate to control information flow from the encoder to the decoder. **Pointer-generator** (See et al., 2017) is an integrated pointer network and seq2seq model. We implemented this baseline without its coverage mechanism since this is not our focus. Baseline models also include two pointer-generator based extensions (Guo et al., 2018; Li et al., 2018). **CGU** (Lin et al., 2018) sets a convolutional gated unit and self-attention for global encoding.

5 Results and Analysis

The following analysis focuses on investigating whether our model is, first, able to generate abstract and new concepts, and, second, how the overall quality performs against the baselines.

5.1 Quantitative Analysis

The results are presented in Table 1. We observe that our model outperformed all the strong state-of-the-art models on both datasets in all metrics except for RG-2 on Gigaword. In terms of the pointer generator performance, the improvements made by our concept pointer are statistically significant ($p < 0.01$) across all metrics.

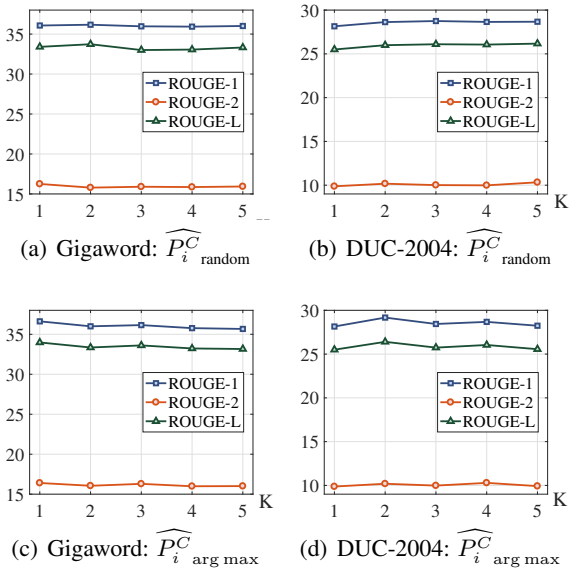
OOV and Summary Length: OOV is another major challenge for current abstractive summarization models. Although generating longer summaries or less UNKs is not our focus, our model still showed improvements in this regard (Table 2). We counted the number of UNKs and all generated summary words and measured the proportions in both testing sets. The OOV percentages dropped from 4.02% to 1.12% on Gigaword and from 2.08% to 0.23% on DUC-2004, which demonstrates that our model is effective at alleviating OOV problems. This result also supports the superior of the concept pointer over the baseline pointer generator. From the statistics, we found that the summaries generated by our concept pointer averaged around 10.46 words, while the pointer generator summaries averaged 10.28 words per summary on DUC-2004. This shows the concept pointer is able to capture more salient content by generating relatively longer summaries.

Abstractiveness: According to Chen and Bansal (2018), abstractiveness scores are computed as the percentage of novel n-grams in the generated summaries that are not included in the source documents. As shown in Table 3, compared with human-written summaries which receive the highest novelty in terms of abstractiveness, our concept pointer generator achieves closest performance with human-written summaries against the baseline. This result demonstrates a further advantage of our model in producing new and abstract concepts. Our model is designed to improve semantic relevance and promote higher abstraction. More generated summary examples can be found in Appendix B.

5.2 Analysis on Training Strategies

To evaluate the relative impact of each training strategy with the model, we tested different combinations for comparison with each other and against the baselines.

Context-aware Conceptualization: To investigate the impact of training with both the number of concepts k and the concept update strategy mentioned in Eqs.(6) and (7), we chose a different number of concept candidates, i.e., $k = 1, 2, 3, 4, 5$, to for the context-aware conceptualization update strategy. Performance was fully



(a) Gigaword: $\widehat{P_i^C}_{\text{random}}$ (b) DUC-2004: $\widehat{P_i^C}_{\text{random}}$

(c) Gigaword: $\widehat{P_i^C}_{\arg\max}$ (d) DUC-2004: $\widehat{P_i^C}_{\arg\max}$

Figure 3: ROUGE metrics on Gigaword and DUC-2004 w.r.t a different number of concept candidates. Updates were conducted by hard assignment $\widehat{P_i^C}_{\arg\max}$ and random selection $\widehat{P_i^C}_{\text{random}}$.

evaluated with the three ROUGE metrics as shown in Figure 3. The results only vary slightly according to the number of concepts with the random selection strategy (Eq.(7)), as shown in Figure 3(a) and 3(b). This indicates that a random strategy is not very sensitive to the number of extracted topics. This is, in part, because the concept pointer may or may not be able to point to the correct concepts from multiple candidates. While in Figure 3(c) and 3(d), the optimum settings are clearly apparent, i.e., $k = 1$ on Gigaword and $k = 2$ on DUC-2004. Overall, the hard assignment strategy (Eq.(6)) provided the best performance in practical terms, while random selection (Eq.(7)) performs stably with different settings.

Training with DS vs. RL: As shown in Table 1, our model with either a distant supervision strategy (concept pointer+DS) or reinforcement learning (concept pointer+RL) were both superior to the basic concept pointer generator on both datasets. Further, the relative improvement of the concept pointer+DS over the concept pointer+RL ranged from 3.5% to 9.6% on DUC-2004 but was inferior to concept pointer+RL on Gigaword. In comparing the results, it is clear that DS training has a noticeable effect when the testing set is substantially semantically different from the training set but provides less improvement than RL when the two are close. From this analysis, we conclude that the DS strategy is better for model adaption with abstractive summarization.

Table 4: Human evaluation: scoring of three models in terms of abstraction and overall quality by human evaluators (the higher the better). The score range could be 0-20. $\star$ indicates the improvements from the baselines to the concept pointer are statistically significant.

Method	abstraction	overall quality
seq2seq+att $\star$	5.85	5.65
Pointer generator $\star$	8.95	8.10
Concept pointer	10.00	9.60

5.3 Human Evaluations

To explore the correctness of our model using human judgment, we conducted a manual evaluation with 20 post-graduate volunteers. We primarily used the following criteria to assess the generated summaries: *abstraction*, i.e., Are the abstract concepts contained in the summary appropriate?; and *overall quality*, i.e., Is the summary readable, informative, relevant, etc.? To conduct the evaluation, we randomly selected 20 examples from the DUC 2004 testing set and asked the volunteers to subjectively assess the summaries. Each example consisted of an article and three summaries, i.e., a summary by the seq2seq model, the pointer generator model, and our proposed concept pointer model. The volunteers chose the best summaries for each of the articles according to the above criteria (can be multiple choices). Obviously, the summaries were randomly shuffled, and the model used to produce each was unknown to prevent bias. The scores for each model were ranked by how many times the volunteers chose a summary w.r.t each criteria, averaged by the number of participants. The results are presented in Table 4, which show that our model outperformed both the seq2seq model and the pointer generator (See et al., 2017) in both criteria.

As a last step, we manually inspect the summaries generated by our model, and some examples are presented in Appendix B. We found that the summaries were not as abstract as human-written summary would likely be. The overarching tendency of the model is still to copy segments of the source text and rearrange the phrases into a summary. However, the overall approach does produce more high-level concepts with correct relations compared to the baselines, which demon-

strates that our solution is a promising research direction to further pursue. Additionally, the generated summaries are long, fluent, and informative.

6 Conclusion

This paper presents a novel concept pointer generator model to improve the abstractive summarization model and generate concept-oriented summaries. We also propose a novel distant supervision strategy for model adaption to different datasets. Both empirical and subjective experiments show that our model makes a statistically significant quality improvement over the state-of-the-art baselines on two popular datasets.

Acknowledgement

This work was supported by National Natural Science Foundation of China (No.61602036, 61751201), partially supported by the Research Foundation of Beijing Municipal Science & Technology Commission (No. Z181100008918002), and partially supported by Ministry of Education China Mobile Research Foundation (No. MCM20170302).

References

Lidong Bing, Piji Li, Yi Liao, Wai Lam, Weiwei Guo, and Rebecca J. Passonneau. 2015. Abstractive multi-document summarization via phrase selection and merging. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL, Volume 1: Long Papers*, pages 1587–1597.

Asli Çelikyilmaz, Antoine Bosselut, Xiaodong He, and Yejin Choi. 2018. Deep communicating agents for abstractive summarization. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, Volume 1 (Long Papers)*, pages 1662–1675.

Yen-Chun Chen and Mohit Bansal. 2018. Fast abstractive summarization with reinforce-selected sentence rewriting. *arXiv preprint arXiv:1805.11080*.

Sumit Chopra, Michael Auli, and Alexander M. Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 93–98.

Bonnie Dorr, David Zajic, and Richard Schwartz. 2003. Hedge trimmer: A parse-and-trim approach to headline generation. In *Proceedings of the HLT-NAACL 03 on Text summarization workshop-Volume 5*, pages 1–8. Association for Computational Linguistics.

Yang Gao, Yang Wang, Luyang Liu, Yidi Guo, and Heyan Huang. 2019. Neural abstractive summarization fusing by global generative topics. *Neural Computing and Applications*.

Pierre-Etienne Genest and Guy Lapalme. 2011. Framework for abstractive summarization using text-to-text generation. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, pages 64–73.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1631–1640.

Han Guo, Ramakanth Pasunuru, and Mohit Bansal. 2018. Soft layer-specific multi-task summarization with entailment and question generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Volume 1: Long Papers*, pages 687–697.

Yidi Guo, Heyan Huang, Yang Gao, and Chi Lu. 2017. Conceptual multi-layer neural network model for headline generation. In *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data*, pages 355–367. Springer.

Wan-Ting Hsu, Chieh-Kai Lin, Ming-Ying Lee, Kerui Min, Jing Tang, and Min Sun. 2018. A unified model for extractive and abstractive summarization using inconsistency loss. *arXiv preprint arXiv:1805.06266*.

Wojciech Kryscinski, Romain Paulus, Caiming Xiong, and Richard Socher. 2018. Improving abstraction in text summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1808–1817.

Haoran Li, Junnan Zhu, Jiajun Zhang, and Chengqing Zong. 2018. Ensure the correctness of the summary: Incorporate entailment knowledge into abstractive sentence summarization. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018*, pages 1430–1441.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*.

Junyang Lin, Xu Sun, Shuming Ma, and Qi Su. 2018. Global encoding for abstractive summarization. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL, 2018, Volume 2: Short Papers*, pages 163–169.

Fei Liu, Jeffrey Flanigan, Sam Thomson, Norman Sadeh, and Noah A Smith. 2018. Toward abstractive summarization using semantic representations. *arXiv preprint arXiv:1805.10399*.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015*, pages 1412–1421.

Ramesh Nallapati, Bowen Zhou, Cícero Nogueira dos Santos, Çaglar Gülçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL*, pages 280–290.

Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2011. English gigaword fifth edition ldc2011t07. dvd. *Philadelphia: Linguistic Data Consortium*.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 379–389.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Volume 1: Long Papers*, pages 1073–1083.

Fei Sun, Peng Jiang, Hanxiao Sun, Changhua Pei, Wenwu Ou, and Xiaobo Wang. 2018. Multi-source pointer network for product title summarization. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 7–16. ACM.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*, pages 2692–2700.

Zhongyuan Wang and Haixun Wang. 2016. Understanding short texts. In *the Association for Computational Linguistics (ACL) (Tutorial)*.

Zhongyuan Wang, Haixun Wang, Ji-Rong Wen, and Yanghua Xiao. 2015. An inference approach to basic level of categorization. In *Proceedings of the 24th acm international on conference on information and knowledge management*, pages 653–662. ACM.

Noah Weber, Leena Shekhar, Niranjan Balasubramanian, and Kyunghyun Cho. 2018. Controlling decoding for more abstractive summaries with copy-based networks. *arXiv preprint arXiv:1803.07038*.

Qingyu Zhou, Nan Yang, Furu Wei, and Ming Zhou. 2017. Selective encoding for abstractive sentence summarization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Volume 1: Long Papers*, pages 1095–1104.

Qingyu Zhou, Nan Yang, Furu Wei, and Ming Zhou. 2018. Sequential copying networks. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18)*, pages 4987–4995.